Characterizing Building Blocks

Metrics for informed design

Matt Reilly

Chief Engineer

SiCortex, Inc

SiCortex

The Problem

- Develop useful system characterization techniques to inform application development.
- Use portable techniques.
- Explore issues of balance.
- Squeeze more performance out of the technology we've got.

An Apology:

- YOU already know all this stuff
- Your colleagues may not
- Use these slides as material for *their* enlightenment.

The Counter Argument

- But if I develop the application with a particular machine model in mind, it will become machine specific!

- Is it better to develop a code that runs equally poorly on *all* machines?
- Pick a model/architecture/vendor that has a point-of-view that aligns with your application domain.

The Computational Model

- For a large set of interesting problems

$$T_{sol} = T_{arith}/N + T_{mem}/N + T_{IO} + f(N)T_{comm}$$

or

$$T_{sol} = MAX(T_{arith}/N, T_{mem}/N, T_{IO}, f(N)T_{comm})$$

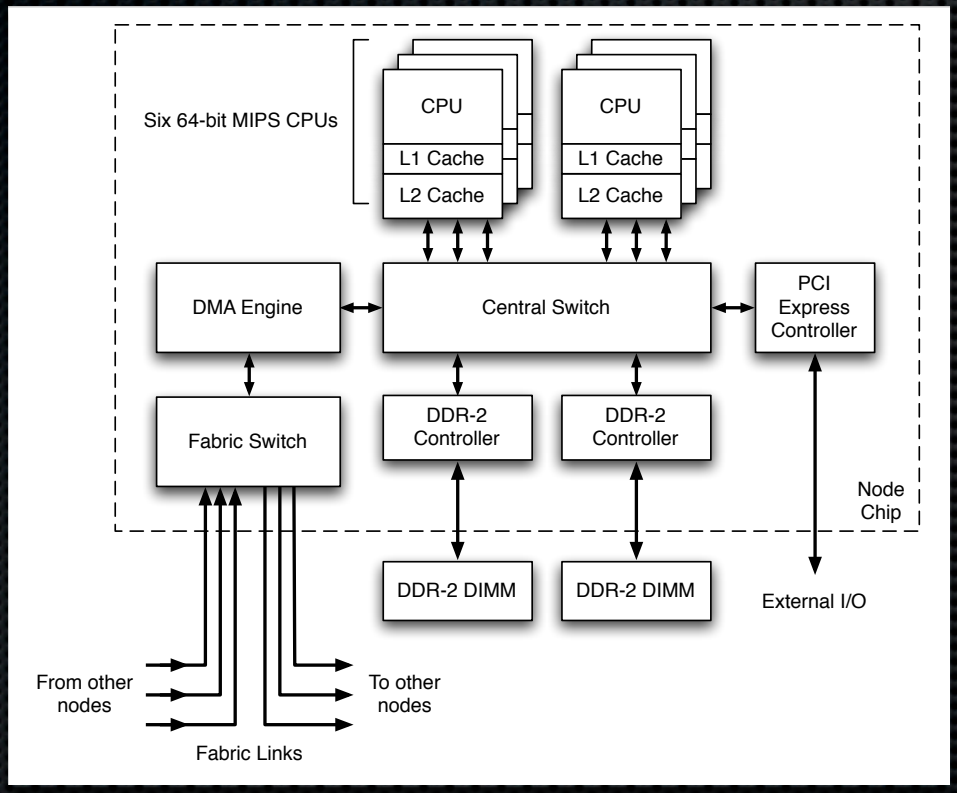
- We know how to characterize T_{arith} (DGEMM is OK)
- Stream Triad is a good indicator for T_{mem}
- How to characterize T_{comm}?

The Platform

- 5832 Low Power MIPS Processors
- 972 Six-way SMP nodes
- High Bandwidth Inter-node Fabric
- 108 PCI Express IO ports
- 72 GB Ethernet
- 15KW to 20KW

SiCortex

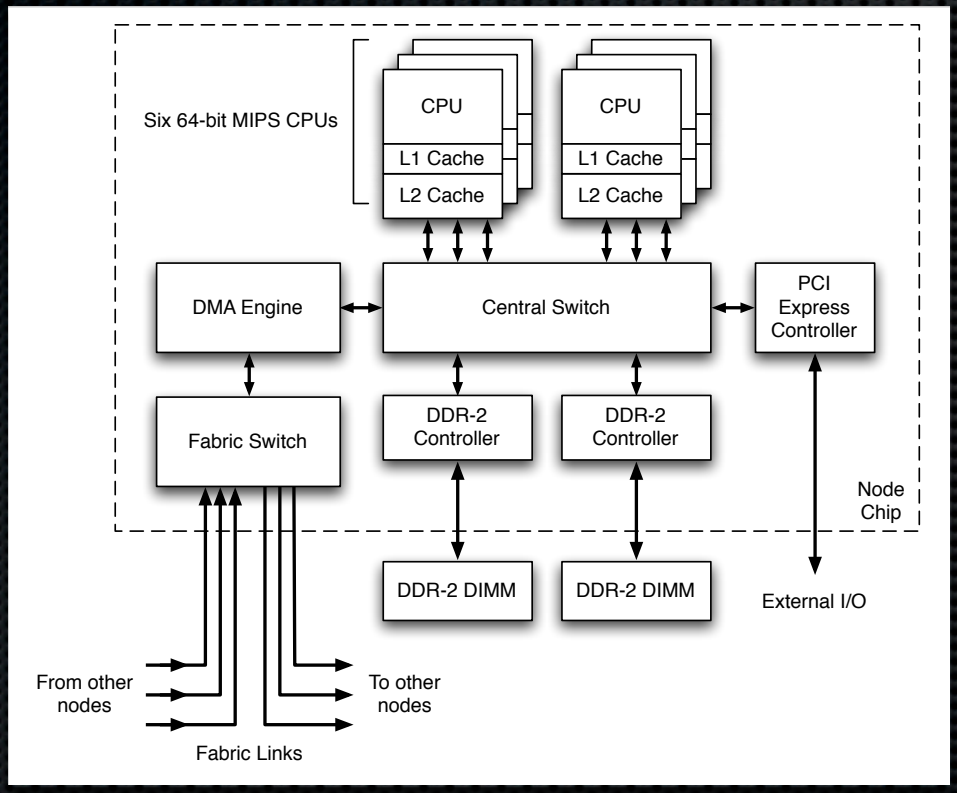
The SiCortex Node Chip



Six way Linux SMP with 2 DDR ports, PCI Express, Message controller, and fabric switch

SiCortex

The SiCortex Node Chip



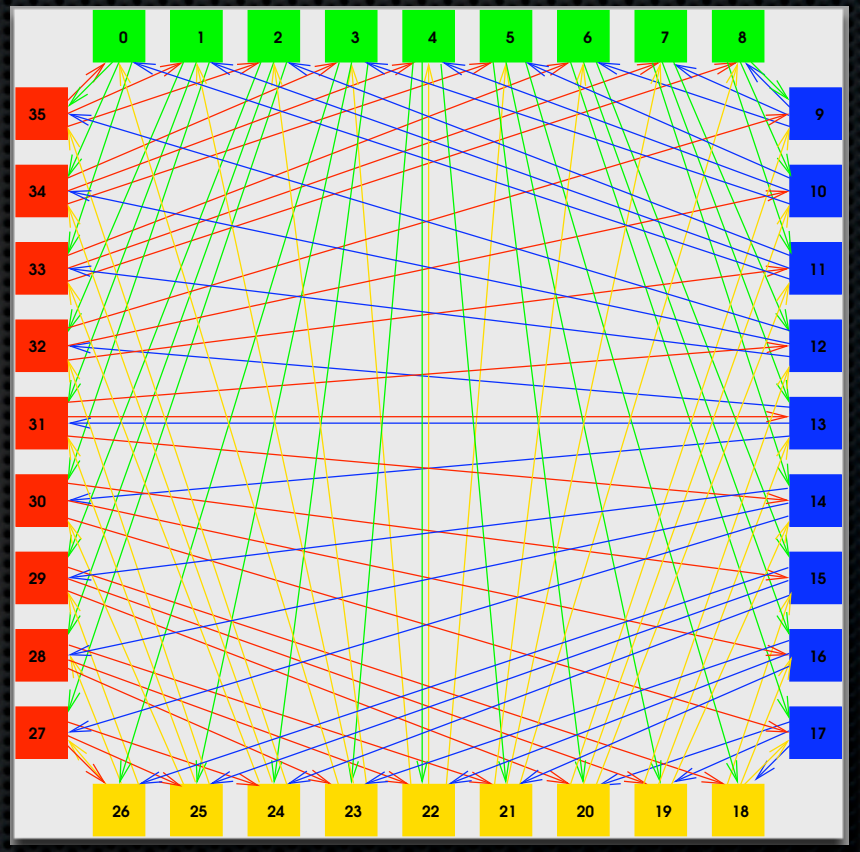
x 972

SC5832: 972 SiCortex nodes connected in a degree 3 Kautz Graph

SiCortex

The Kautz Graph

- Logarithmic diameter
- Reconfigure around failures
- Very fast collectives
- 972 nodes: diameter 6



SiCortex

The Computational Model

- For a large set of interesting problems

$$T_{sol} = T_{arith}/N + T_{mem}/N + T_{IO} + f(N)T_{comm}$$

or

$$T_{sol} = MAX(T_{arith}/N, T_{mem}/N, T_{IO}, f(N)T_{comm})$$

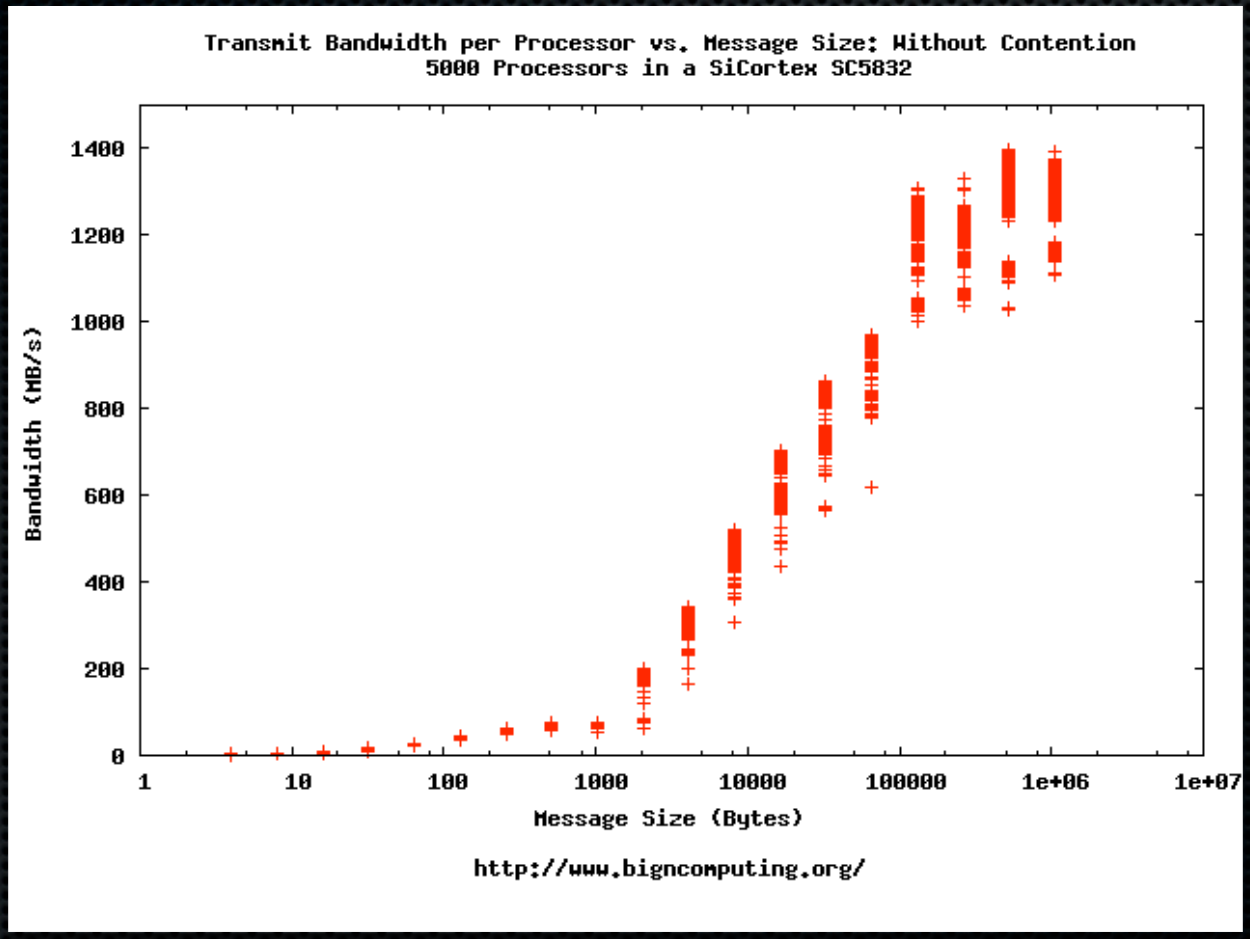
- How to characterize T_{comm}?

MicroBenchmarks and Kernels

- MPI Latency - 1.4 µsec
- MPI BW - 1.5 GB/s
- HPC Challenge work underway
- SC5832, on 5772 cpus:
  - DGEMM 72%
  - HPL 3.6 TF (83% of DGEMM)
  - PTRANS 210 GB/s
  - STREAM 345 MB/s (1.9 TB/s aggregate)
  - FFT 174 GF
  - RandomRing 4 usec, 50 MB/s
  - RandomAccess 0.74 GUPS (5.5 optimized)

SiCortex

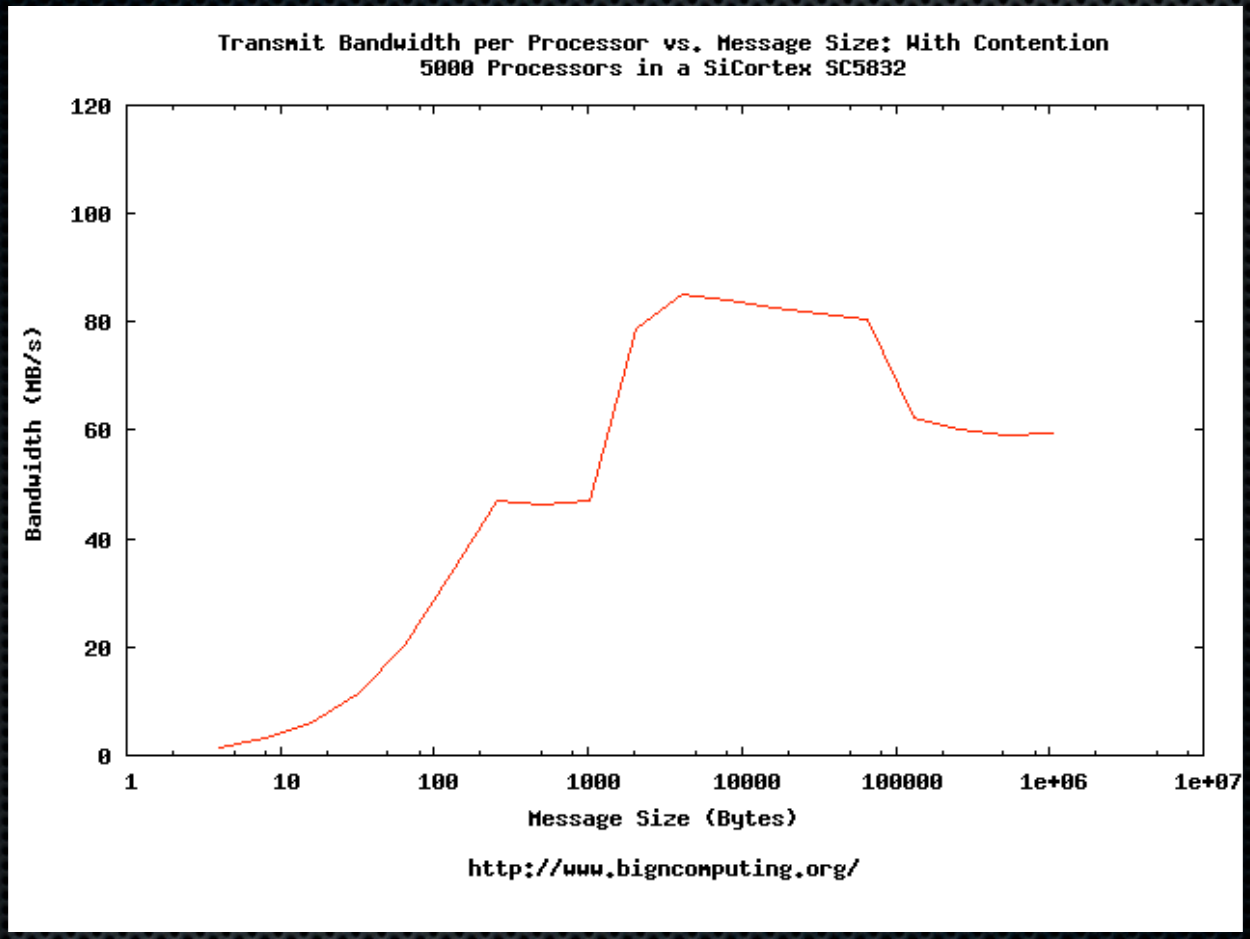
Zero contention message bandwidth?



Interesting relationship between message size and bandwidth

SiCortex

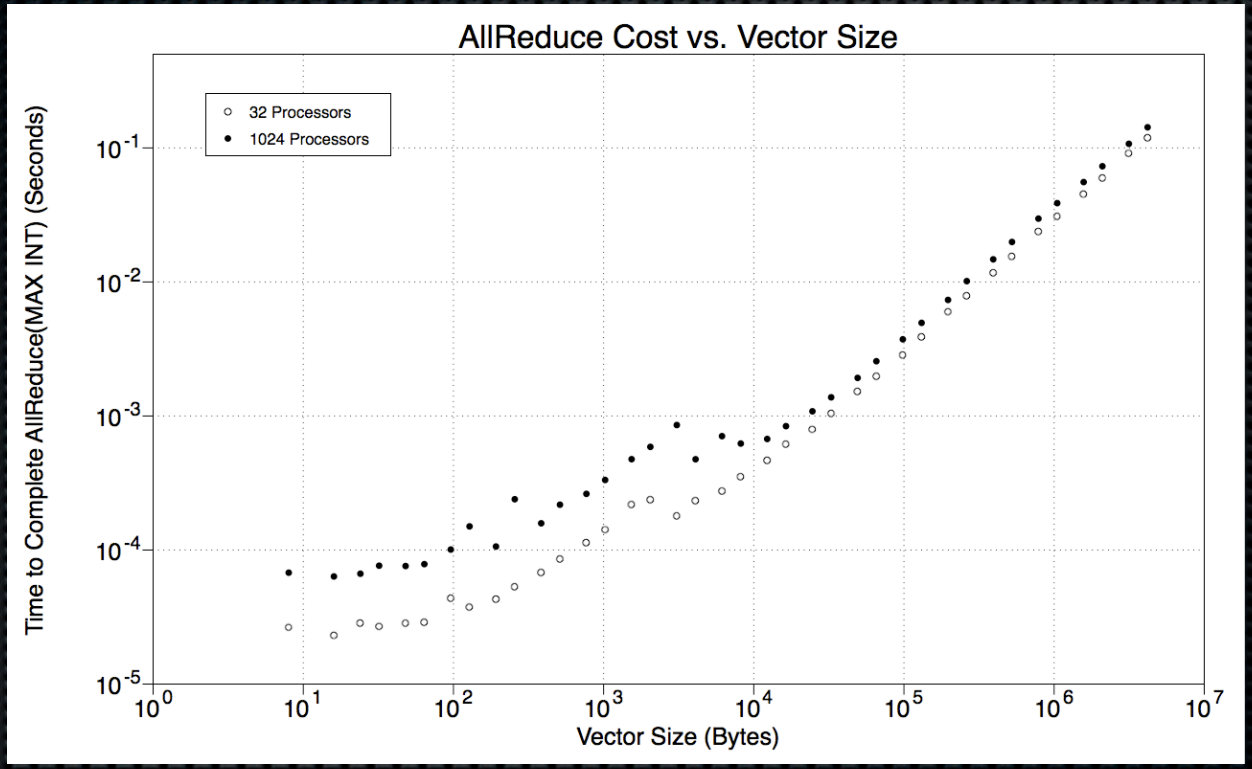
Communication in “real world” conditions



Contention matters. (For more, see Abhinav Bhatele’s work at http://charm.cs.uiuc.edu/ .)

SiCortex

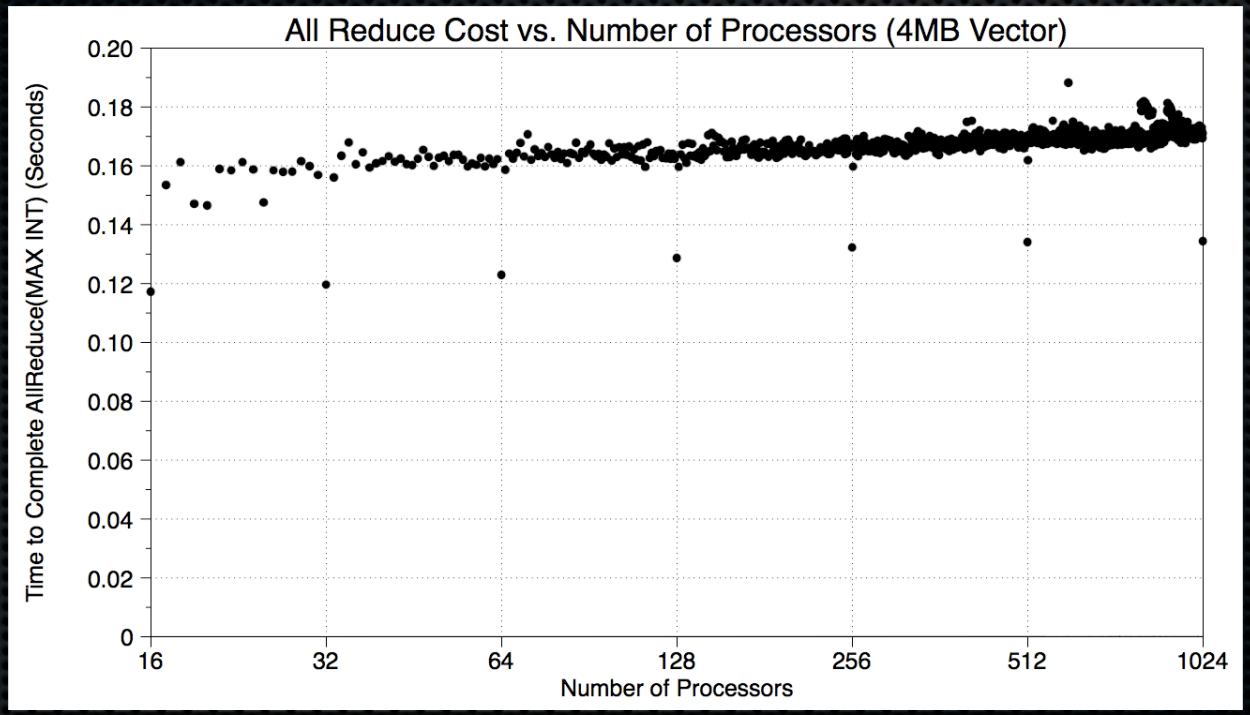
What about Collectives?



Dependence on vector size is predictable.

SiCortex

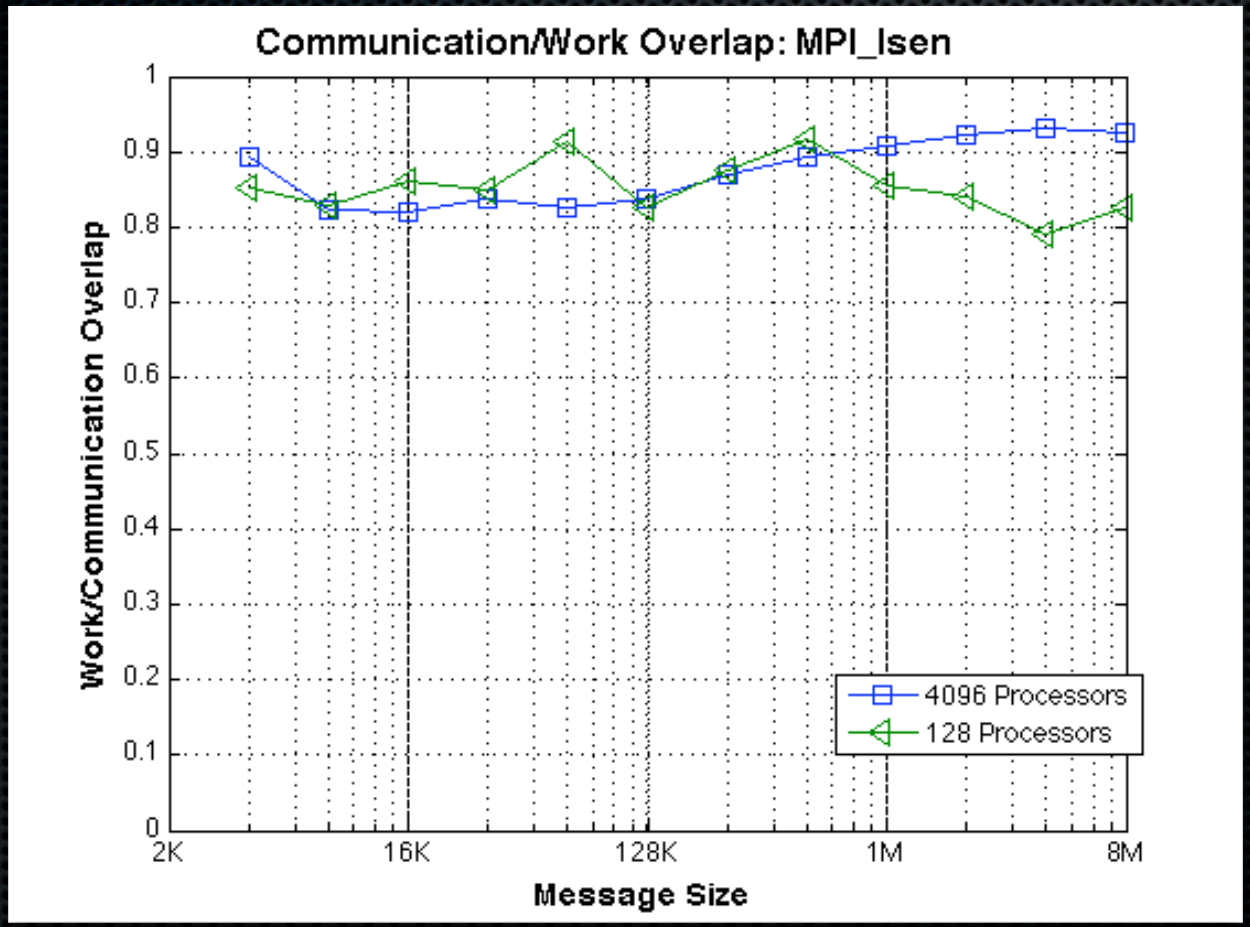
Collective Cost vs. Number of Processes



Note the optima.

SiCortex

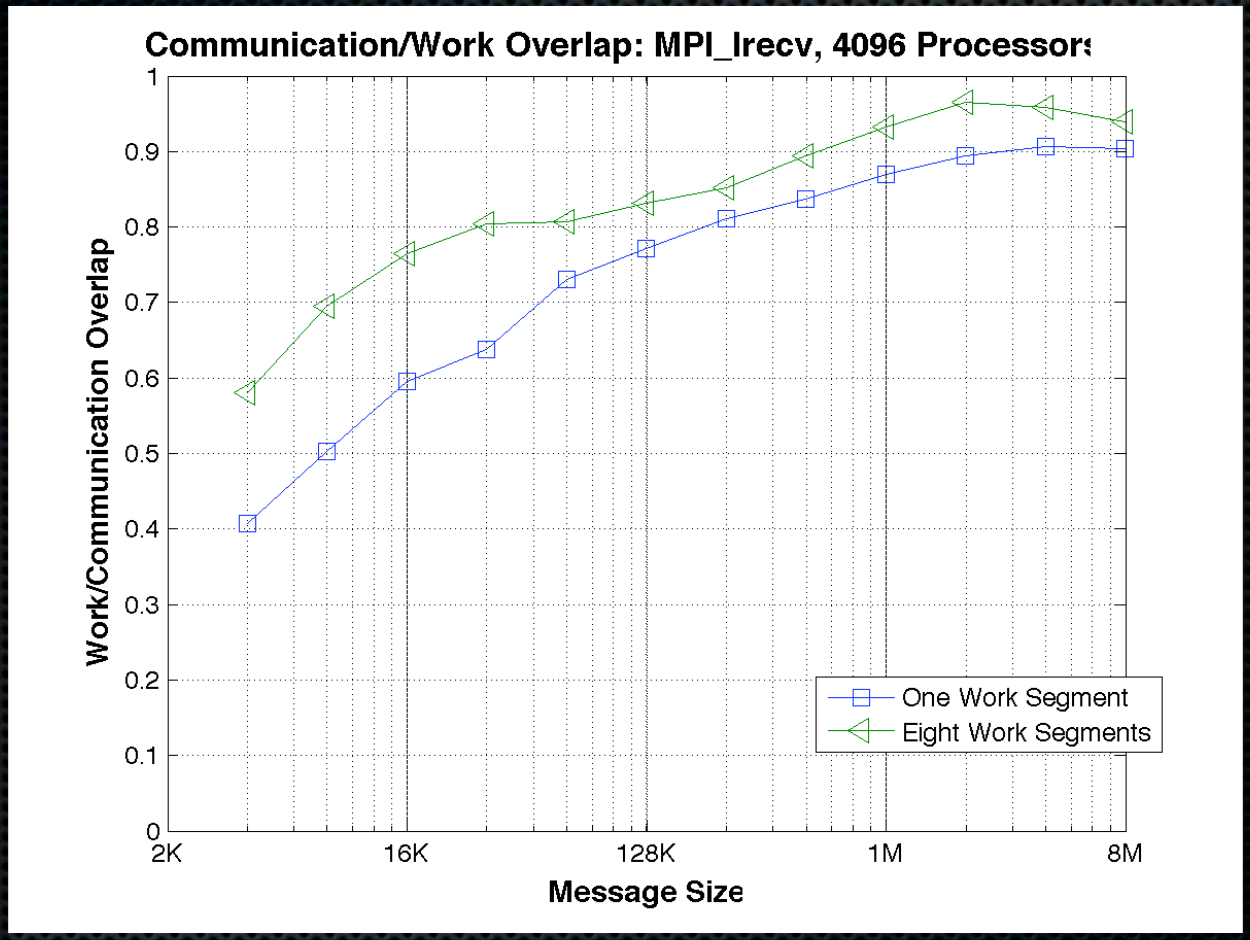
How much can we overlap “work” with “comm”?



Can we mitigate T_{comm}?

SiCortex

How much Overlap with Receive Ops?



It may pay to fiddle the algorithm a bit...

SiCortex

Key MPI Performance Features

- Sweet spot is for messages from 2KB to 64KB.
- Best high-contention BW is around 80MB/s.
- Simple collectives cost
  - < 200nS per byte for 1K to 10K.
  - < 60nS per byte for 10K and up.
- Good send/recv overlap opportunities for messages > 16kB

Some Examples

- TeraByte Sort
- Three-Dimensional FFT

TeraByte Sort

- Sort 10 billion 100 byte records (10 byte key).
- Leave out the IO (this isn't quite the Indy TeraSort benchmark)
- Use 5600 processors

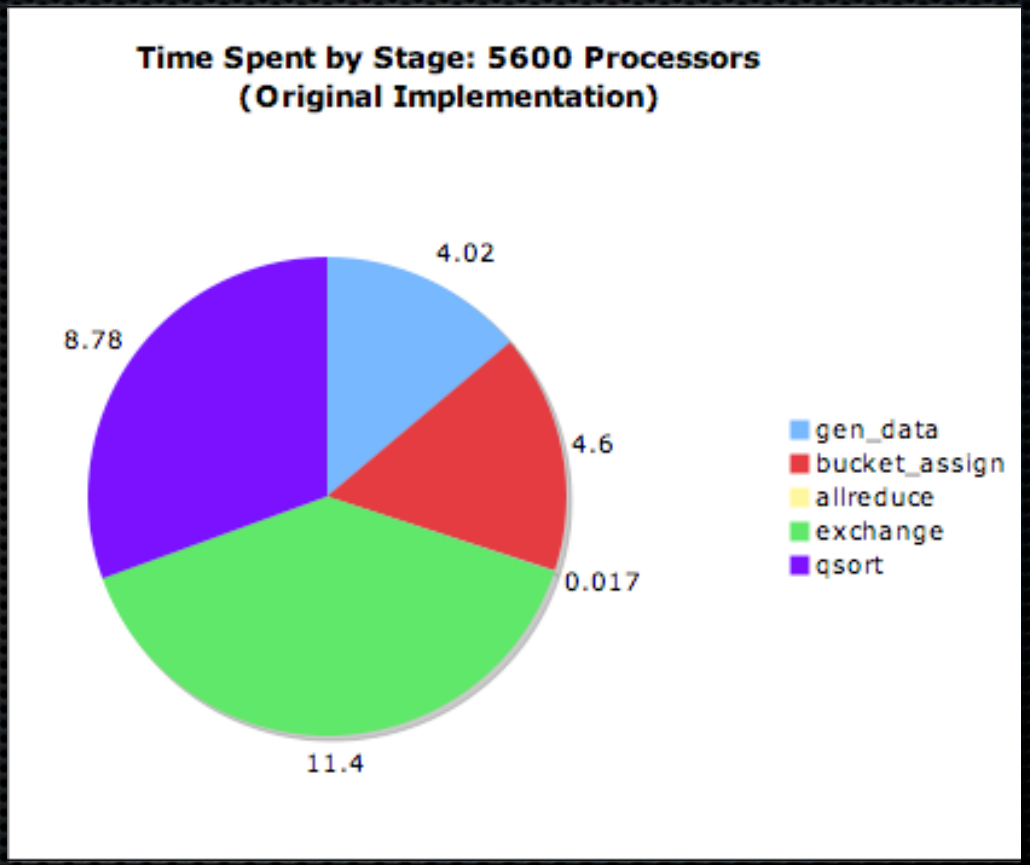
- Key T_{comm} attributes:
  - Time to exchange all 1TB is about 4 sec +/-
  - Time to copy each processor's sublist is about 1 sec +/-
  - Global AllReduce for a 256KB vector is O(10mS)

Problem Approach

- Use Bucket/Radix Sort to partition list among workers
  - bucket assignment (copy sublist and re-arrange) 1sec+
  - global All Reduce: 0 sec +
  - complete exhange of all records 4 sec+
  - parallel sort of ~2M records 4 sec+ (42M record accesses that cause a cache miss... T_{mem} is often proportional to main memory access time.)

From first implementation...

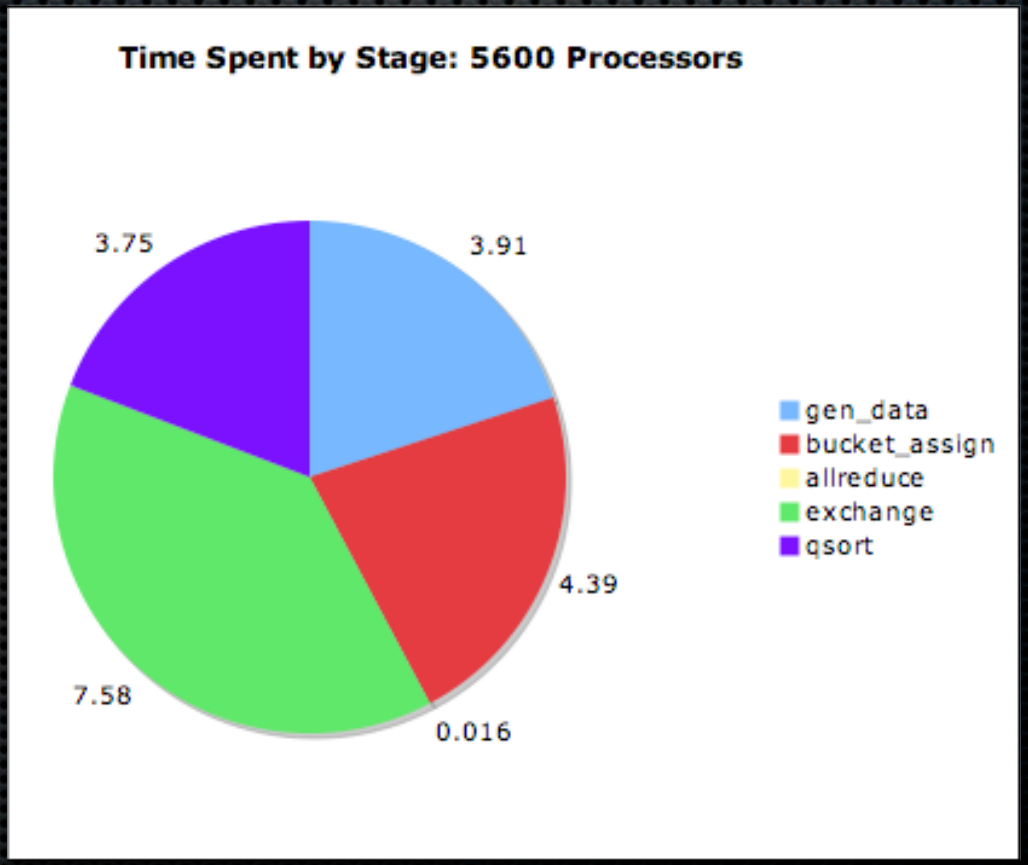
- Exchange seems a little long.
- QSort is way too long.
- Bucket assign seems long.
- Use our initial model to guide the tuning effort.



SiCortex

Tuning....

- Improved QSort to the model target
- Bucket assignment is still very slow
- Exchange is still a little slow (by almost 2x!)
- We can do better...



SiCortex

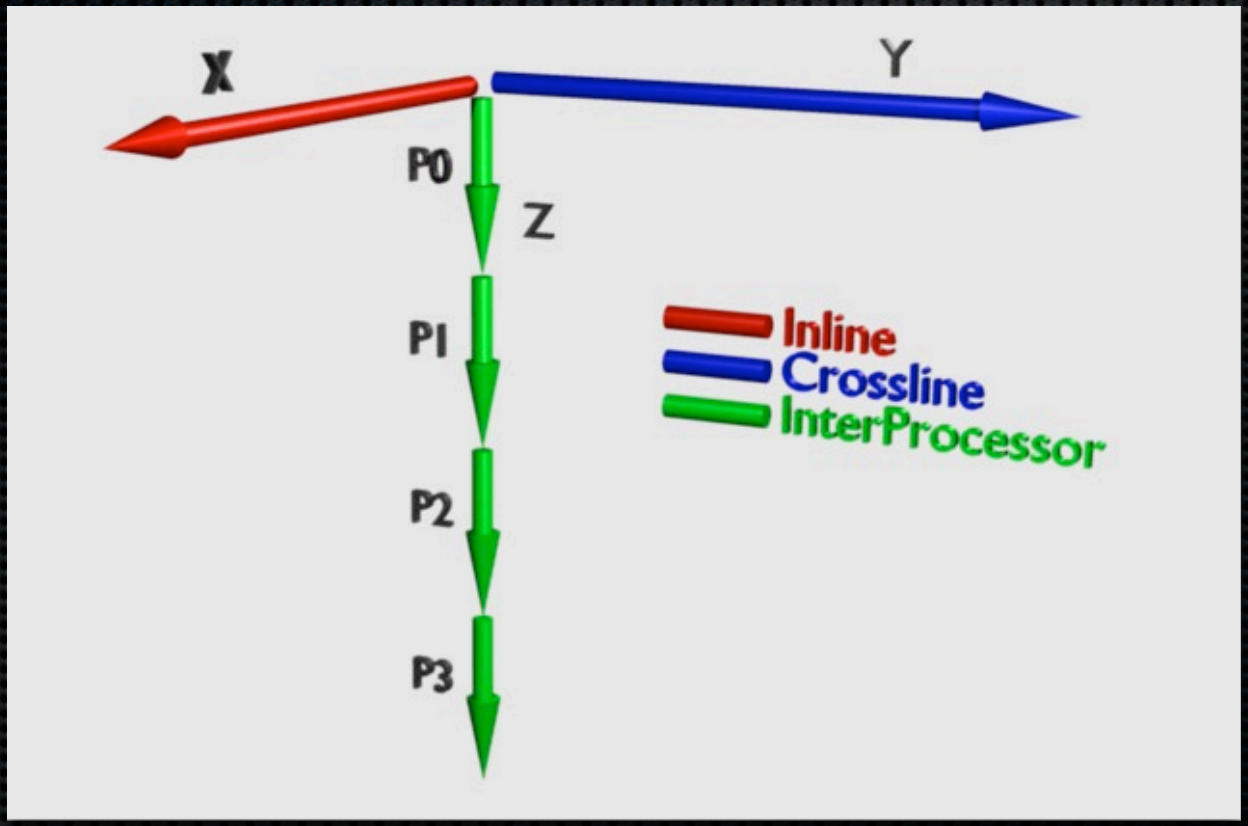
Three-Dimensional FFT

- 3D FFT of 1 billion point volume
- Use PFAFFT (prime factor analysis)
  - complex-complex single precision
  - 1040 x 1040 x 1040
- Two target platforms:
  - SC072 -- 72 processors
  - SC1458 -- 1458 processors

Problem Approach

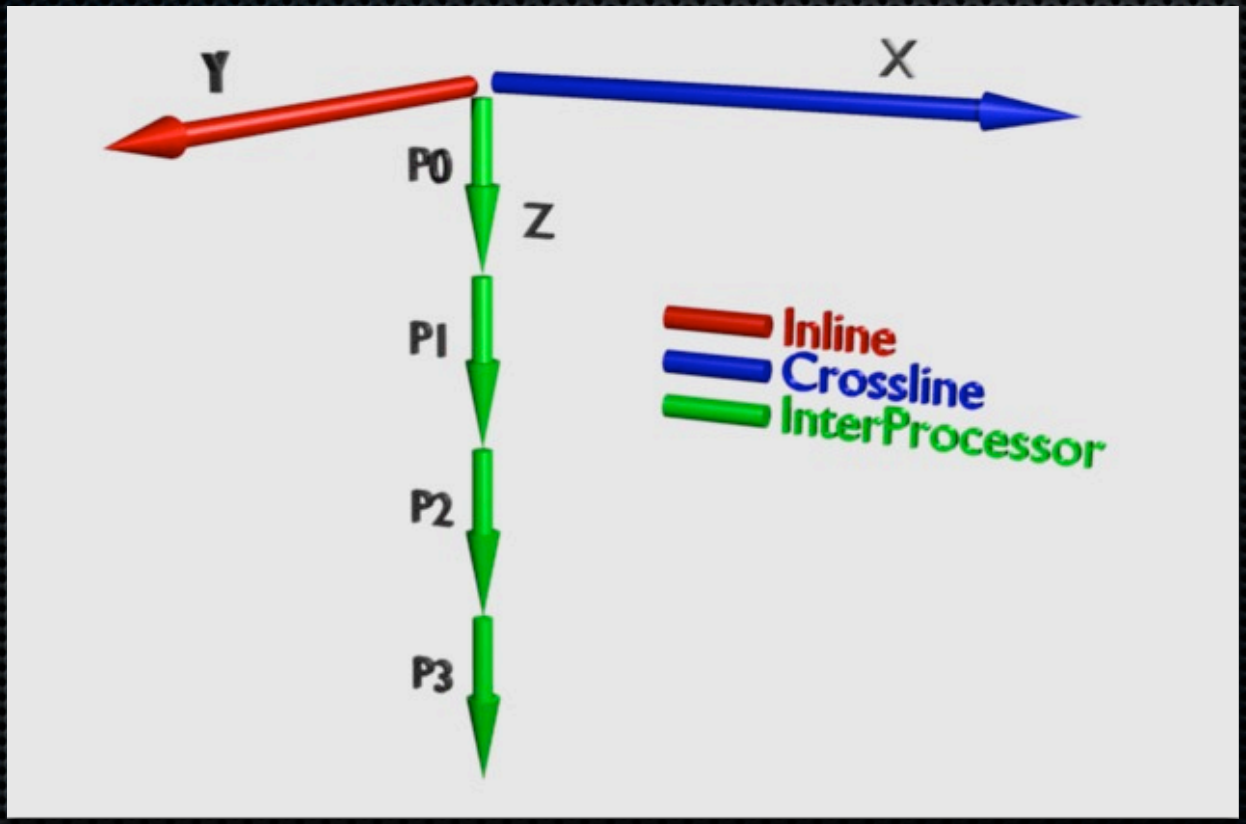
- We know T_{arith} for 1040 point FFT ~ 200uS
- 1040 = 5 * 13 * 16: so
  - SC072: use 65 processors
  - SC1458: use 1040 processors
- Concentrate on the 65 processor implementation
  - Each proc owns 16 planes of 1040 x 1040
  - Each proc performs 16 2D FFT ops (about 6 sec)
  - Global 3D transpose: (128MB per proc, about 1.5 sec)
  - Each proc performs 16 *1040 1D FFT ops (about 3 sec)

Step 1: FFT in X direction



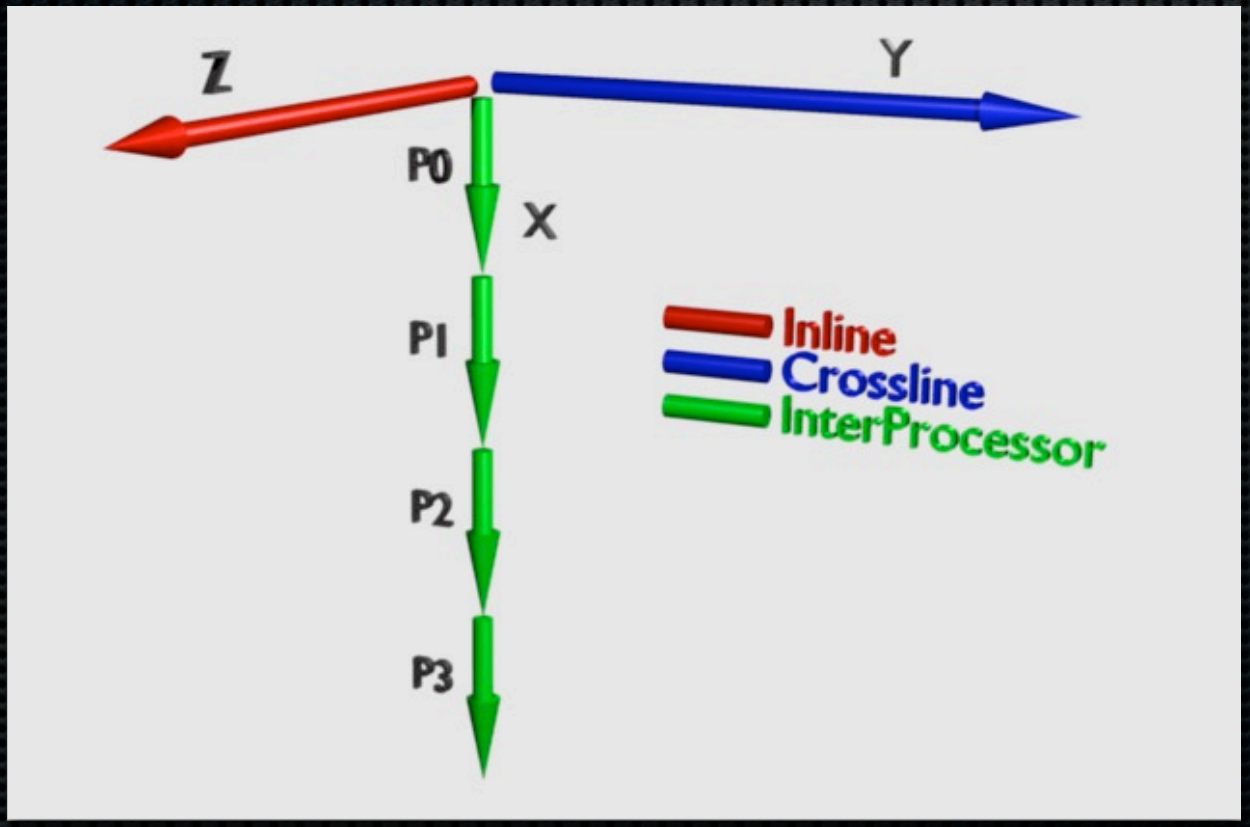
SiCortex

Step 2: FFT in Y direction



SiCortex

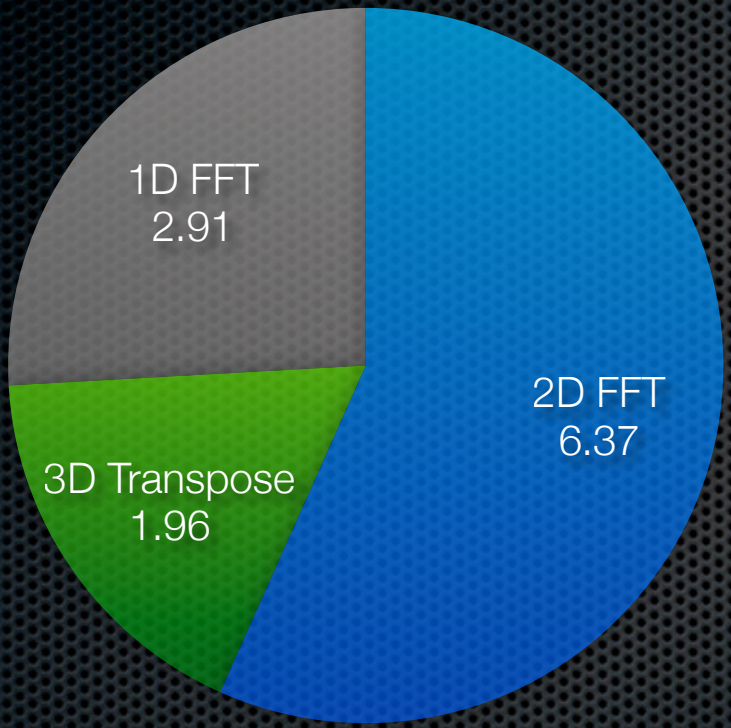
Step 3: FFT in Z direction



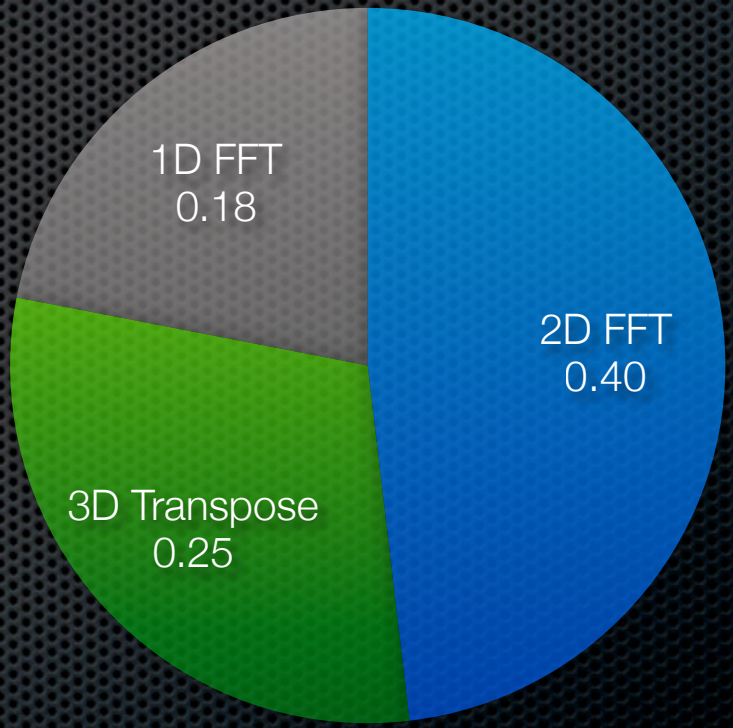
SiCortex

Results!

65 Processor 3D FFT



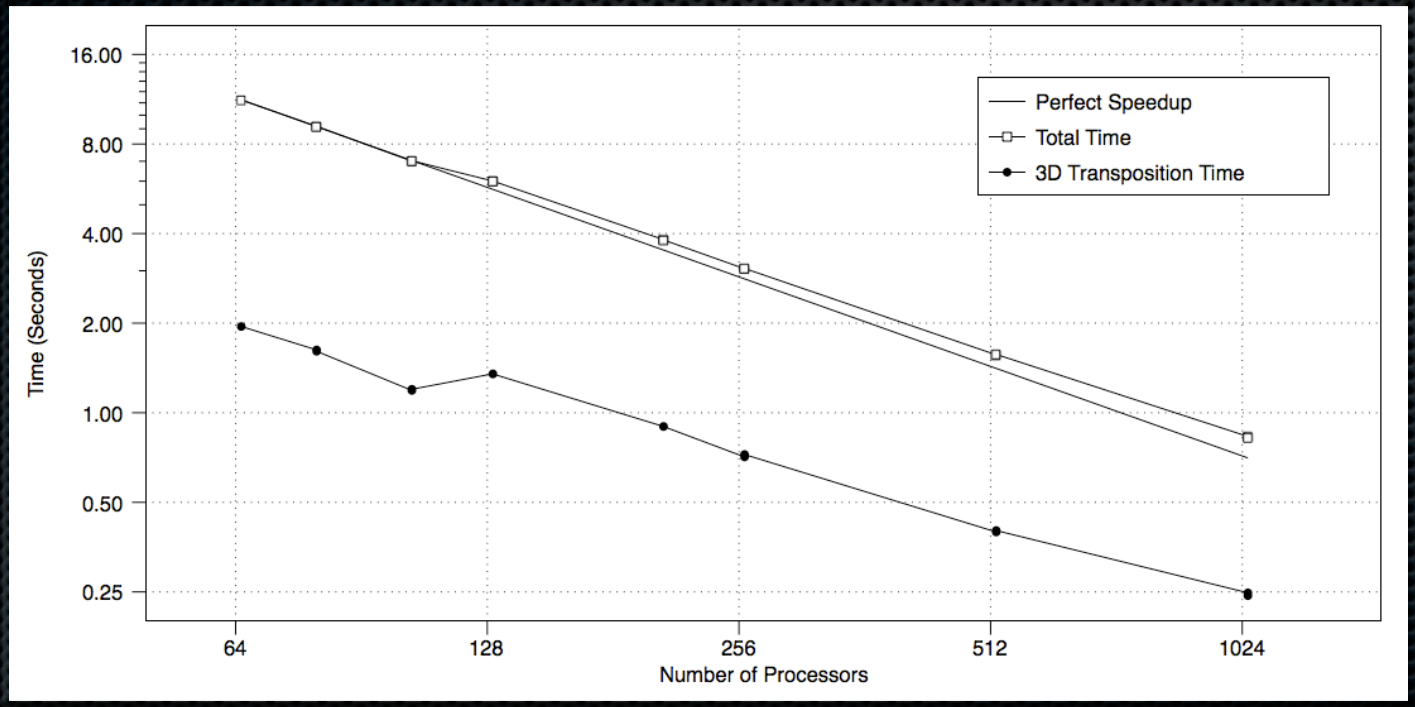
1040 Processor 3D FFT



Prediction was pretty much on target. (But the transpose is still a little slow...)

SiCortex

Scaling on the SC1458



The transpose bump is interesting...

SiCortex

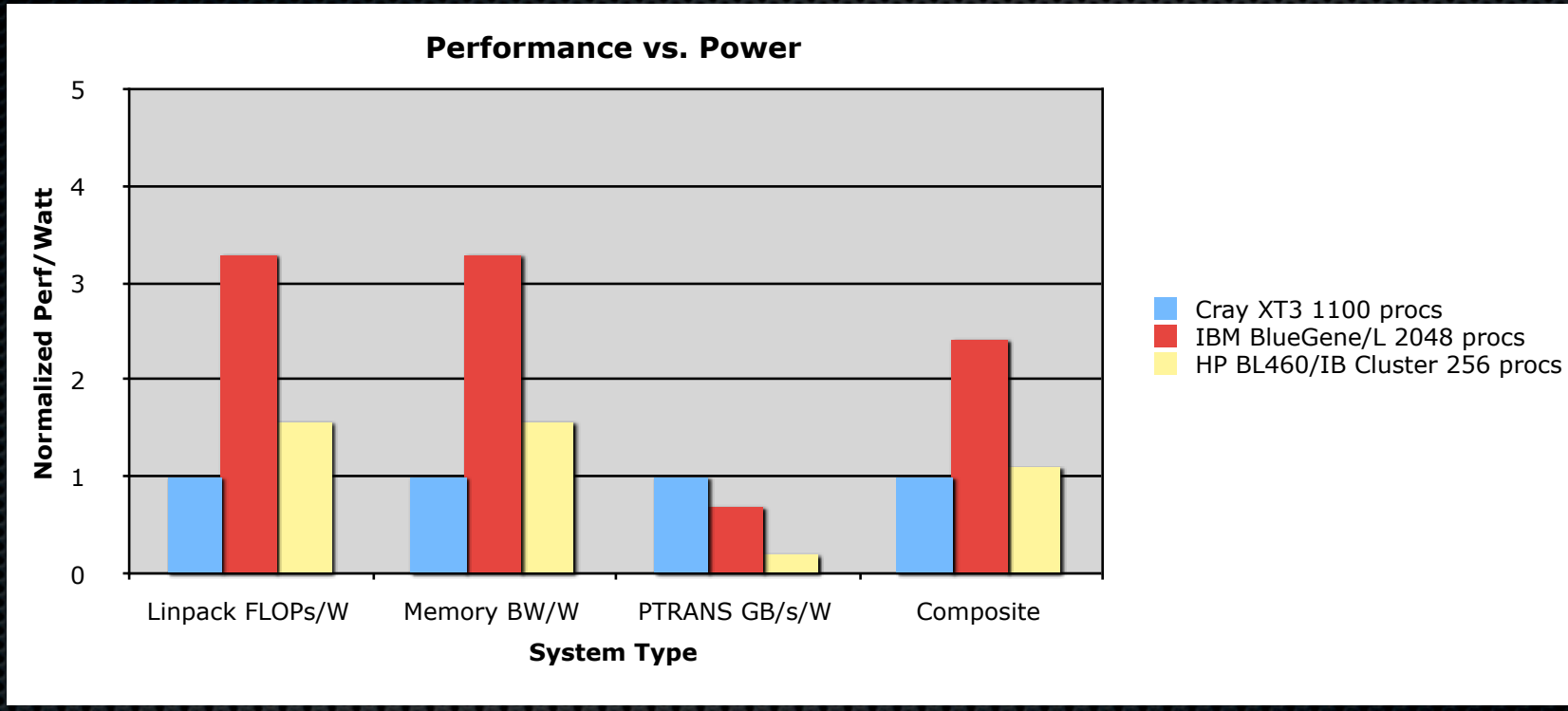
If there is time for a Rant:

- Our approach to “Green” characterization so far is...:
  - Too often vendors “paint it green”
  - Closer in shade to a pale chartruese
- If performance is N-dimensional, we should measure “Green-ness” that way.
  - Linpack/Watt assumes that all computers spend all their time multiplying large matrices.
  - We should recognize the tuple.

A Green Evaluation Model

- Tuple: <DGEMM, STREAM-TRIAD, G-PTRANS> to cover T_{arith}, T_{mem}, T_{comm}
- Normalize wrt a generally respected model:
  - Cray XT3 - well balanced, not quirky
- Measure Tuple, divide by power input (+ cooling cost?), normalize to XT3

A Comparison



SiCortex

Wrap-up

- Simple metrics can successfully guide *small* development efforts.
  - can this work in the large?
- T_{comm} is properly considered as a set of values, not a scalar.
- It may be that systems can be *usefully* characterized by a small tuple.
  - <DGEMM, PTRANS, STREAM-TRIAD> ?

The Commercial Message

And check out
http://www.BigNComputing.org/

SiCortex

Many of my other computers are

SiCortex